Conducting Repeatable Experiments in Highly Variable Cloud Computing Environments

Ali Abedi
University of Waterloo
ali.abedi@uwaterloo.ca

Tim Brecht
University of Waterloo
brecht@cs.uwaterloo.ca

ABSTRACT

Previous work has shown that benchmark and application performance in public cloud computing environments can be highly variable. Utilizing Amazon EC2 traces that include measurements affected by CPU, memory, disk, and network performance, we study commonly used methodologies for comparing performance measurements in cloud computing environments. The results show considerable flaws in these methodologies that may lead to incorrect conclusions. For instance, these methodologies falsely report that the performance of two identical systems differ by 38% using a confidence level of 95%. We then study the efficacy of the Randomized Multiple Interleaved Trials (RMIT) methodology using the same traces. We demonstrate that RMIT could be used to conduct repeatable experiments that enable fair comparisons in this cloud computing environment despite the fact that changing conditions beyond the user's control make comparing competing alternatives highly challenging.

1. INTRODUCTION

Cloud computing environments [11] like Amazon's EC2 are attractive facilities on which to conduct an experimental performance analysis. Large numbers of machines can potentially be used to compare two or more competing alternative algorithms, designs, or implementations of a system (simply called *alternatives* for the remainder of the paper). Some example alternatives might include, comparing different applications that perform the same function or different versions of the same application. In this case, some experiments may be conducted in parallel on different systems at the same time within different portions of the cloud environment and may also be executed at different times. Additionally, some experiments may not be conducted in parallel but may be conducted by running alternatives during different periods of time.

Unfortunately, performance measurements in cloud computing environments can vary significantly. This may be due to being assigned different physical systems across different experiments at different times. Or, the performance may be impacted due to sharing, in this case variations occur due to other applications that are executing on the same physical hosts (e.g., affecting disk throughput) or elsewhere in the same cloud environment (possibly affecting network latencies or throughput).

When conducting a comparison of two or more competing alternatives, care must be taken to ensure that any differences in performance are due only to the differences in alternatives and not because, one alternative might be executing in a more favorable environment (e.g., in the absence of other applications running on the same node or on a faster node). Therefore, in this paper we examine approaches that can be used to obtain repeatable performance measurements in cloud computing environments which are a prerequisite for comparing the performance of competing alternatives.

In order to conduct a fair comparison of the performance evaluation methodologies, we employ traces collected and analyzed by an independent research group [14]. Schad et al. [14] reported high levels of variation in all studied benchmarks (i.e., for CPU, memory, disk, and network). We utilize these traces to evaluate commonly used methodologies for conducting experimental evaluations in cloud computing environments (refer to Section 4 for more details). We show that some of these techniques might lead to flawed or misleading results despite including a measures of variation such as confidence intervals. Finally, we describe and evaluate a technique called Randomized Multiple Interleaved Trials (RMIT). Although Schad et al. [14] concluded that "naive runtime measurements on the cloud will suffer from high variance and will only be repeatable to a limited extent", we show than RMIT can be used to obtain repeatable and fair comparisons between multiple alternatives for their entire 30 day traces.

The contributions in this paper are:

- We demonstrate that commonly used approaches for comparing competing alternatives have serious flaws and may lead to incorrect conclusions.
- We show that the methodology of using randomized multiple interleaved trials can be used to obtain repeatable empirical measurements in the EC2 environment and therefore is a good candidate as a methodology for comparing the performance of competing alternatives.

We focus on the challenges of conducting repeatable experiments and fair comparisons of multiple alternatives executed at different periods in time. The problem of repeatable and fair comparison of experiments conducted in parallel is discussed in Section 5 and will be the subject of future research.

2. RELATED WORK

Studies examining the variability of measured performance when using the EC2 cloud infrastructure show that the variance on EC2 performance is very high [14, 4, 11, 8]. Schad et al. [14] propose a set of guidelines to reduce the performance variation while conducting experimental evaluations on EC2. Despite the provided

ICPE'17, April 22 - 26, 2017, L'Aquila, Italy

ISBN 978-1-4503-4404-3/17/04...$15.00

DOI: http://dx.doi.org/10.1145/3030207.3030229

guidelines, the authors point out that even with such guidelines, comparing multiple alternatives and conducting repeatable experiments on EC2 might be challenging or impossible due to the high variability. In the paper, we directly examine methodologies for conducting experiments in such environments.

Despite, the high variation in the performance of cloud computing systems, a number of studies [10, 4] choose to run the experiment only once. More rigorously conducted evaluations involve running experiments multiple times and reporting the average value. For example, some studies [7, 12, 5, 6] use an evaluation technique we call multiple consecutive trials (described in Section 3). In Section 4, we show how variations in the cloud environment might lead to misleading results if these techniques are utilized for evaluation.

We recently studied methodologies for conducting repeatable experiments and fair comparisons when performing performance evaluations in WiFi networks [2]. This study [2] shows that many commonly used techniques for the experimental evaluation of WiFi networks are flawed and could result in misleading conclusions being drawn. In that work, although we propose the use of randomized multiple interleaved trials (RMIT) (described in Section 3) as a methodology for coping with changing wireless channel conditions, randomization was not necessary. In this paper, we examine commonly used approaches for measuring and comparing performance as well as the suitability of RMIT in a completely different scenario, namely cloud computing environments. We find that randomization is required, due to periodic changes in the environment, and that RMIT can be used to obtain repeatable results.

Some work [3, 4] has proposed techniques to reduce the variability of application performance when executing in cloud environments by limiting variability of network performance. Unfortunately, such work only reduces but does not eliminate variability. Other shared resources such as disks can also cause performance measurements to be highly variable. One study [9] reports that during off-peak hours disk read bandwidths would range from 100–140 MB/sec, while during peak hours it ranged from 40–70 MB/sec. Moreover, even with techniques to reduce variability, methodologies are still needed to ensure that differences in performance of different alternatives are due to the differences in those alternatives, rather than differences in the conditions under which they were executed.

3. OVERVIEW OF METHODOLOGIES

We use the term *trial* to refer to one measurement, typically obtained by running a benchmark or micro-benchmark for some period of time (the length of the trial). An *experiment* can be comprised of multiple trials executing the same benchmark, where the results of the experiment are reported over the multiple trials (e.g., the average of the measurements obtained over the trials).

Most experiments conducted in cloud computing environments utilize the single trial or multiple consecutive trials methodologies (referred to as *commonly used methodologies* in this paper). In previous work [2], we have argued for and demonstrated the use of Multiple Interleaved Trials and Randomized Multiple Interleaved Trials. We now briefly explain each of these methodologies.

- **Single Trial**: In this approach, an experiment consists of only a single trial. Figure 1-A shows an example of this approach with three alternatives. The performance results obtained from single trials are compared directly. This is the easiest methodology for running an experiment to compare multiple alternatives.
- **Multiple Consecutive Trials**: All trials for the first alternative are run, followed by the second alternative and each of the remaining alternatives. Figure 1-B shows the Multiple Consecutive Trials technique for 3 alternatives.
- **Multiple Interleaved Trials**: One trial is conducted using the first alternative, followed by one trial with the second, and so on until each alternative has been run once. When one trial has been conducted using each alternative we say that one *round* has been completed. Rounds are repeated until the appropriate number of trials has been conducted (Figure 1-C).
- **Randomized Multiple Interleaved Trials**:
  If the cloud computing environment is affected at regular intervals, and the intervening period coincides with the length of each trial, it is possible that some alternatives are affected more than others. Therefore, the randomized multiple interleaved trials methodology randomly reorders alternatives for each round (Figure 1-D). In essence, a randomized block design [13] is constructed where the blocks are intervals of time (rounds) and within each block all alternatives are tested, with a new random ordering of alternatives being generated for each block.

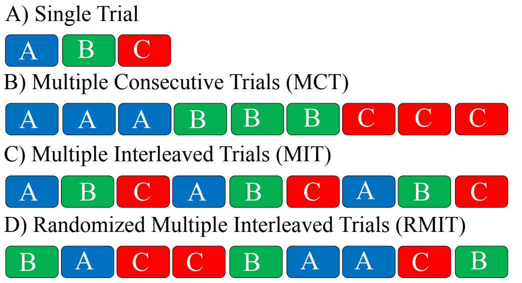

A) Single Trial
A B C
B) Multiple Consecutive Trials (MCT)
A A A B B B C C C
C) Multiple Interleaved Trials (MIT)
A B C A B C A B C
D) Randomized Multiple Interleaved Trials (RMIT)
B A C C B A A C B

Figure 1: Different methodologies: with 3 alternatives

3.1 Methodologies used in Practice

To illustrate that these methodologies are actually used in practice, we studied the performance evaluation methodologies used in the 38 research papers published in the ACM Symposium on Cloud Computing 2016 (SoCC'16). 9 papers conduct experimental evaluations on public clouds (7 on Amazon EC2, 1 on Microsoft Azure, and 1 on Google computing engine). We found that the single trial and multiple consecutive trials (MCT) methodologies are utilized by 7 and 4 papers respectively (some papers use both techniques). No other evaluation methodology is used in these papers. Additionally, 9 other papers also use these two methodologies when conducting evaluations on research clusters.

The fundamental problem is that researchers often incorrectly assume that the characteristics of the systems and networks being used and workloads that are simultaneously executing during their experiments, do not change in ways that impact the performance of the artifacts they are evaluating. Previous work has demonstrated that performance is in fact impacted [14, 11]. In this paper, we examine methodologies commonly used for comparing performance in cloud environments and describe our RMIT methodology that is designed to handle environments in which performance measurements are variable due to circumstances which can not be controlled. To the best of our knowledge, ours is the first work that studies the (randomized) multiple interleaved trials methodologies in the context of the repeatability of experiments in cloud computing environments.

4. EVALUATION

For our evaluation, we utilize traces, collected from benchmark measurements by other researchers [14], conducted over extended period of time on Amazon EC2 servers. The authors of that paper have granted us permission to make the data publicly available [1]. We use these traces to simulate the execution of applications which are utilized by researchers to conduct performance evaluation studies. Note that although these traces were collected in 2010, they are still useful for this study because we do not perform any performance evaluation, instead we evaluate the efficacy of performance evaluation methodologies. Therefore, the age of traces is not very relevant if the performance of cloud computing environments are still variable. A recent study [11] shows that the performance of CPU, memory, and disk benchmarks are highly variable on EC2, specially for I/O-bound applications. The coefficient of variation (CV) reported for the studied benchmarks is as high or even higher (i.e., greater than 0.4) than that of the 2010 study [14].

In the original study [14] a batch of CPU, memory, disk, and network benchmarks are run every hour for one month. The benchmarks used are Unix Benchmark Utility (Ubench) for CPU and memory speed experiments and Bonnie++ for disk I/O experiments Two types of virtual machines used for collecting the measurement data, namely, small and large machine instances. The configurations of these instances are summarized in Table 1. We make use of two different traces collected on servers residing in the United States and in Europe, represented by US and EU, respectively.

Instance	Memory	Cores	Storage	32/64
Small	1.7 GB	1	160 GB	32-bit
Large	7.5 GB	2	850 GB	64-bit

Table 1: Instances configurations

Schat et al. [14] report that EC2 performance varies so much that wall clock experiments must be conducted with "considerable care". Figure 2 shows the CPU benchmark results for large instances located in the US, collected over one month from their study. Each point shows the benchmark score for a single trial (there is one trial per hour). Despite the identical configuration of the virtual machines used for this experiment, the CPU performance varies considerably. The memory and disk performance benchmarks illustrate similar behavior with large variations over the duration of the experiment.

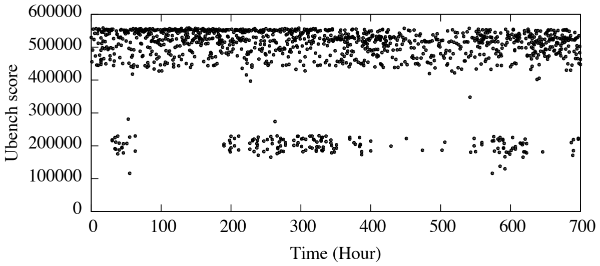


Figure 2: CPU performance, large instance, US [14].

Table 2 summaries the statistical properties of all traces we utilize in this paper. For each combination of benchmark, instance size, and region, we report minimum, maximum, mean, standard deviation (SD), and coefficient of variation (CV). These measures are presented to better understand how measured performance can vary from one experiment to another. The minimum and maximum measured values for all benchmarks show that the benchmark scores cover a wide range relative to mean. For instance, in the CPU benchmark of large instances located in the US, the benchmark score varies from 116,243 to 558,446. The mean and the standard deviation also confirm that CPU benchmark scores are highly variable in this trace. We also present the coefficient of variation (i.e., SD/mean) in order to compare the level of variation across different traces. The coefficient of variation indicates that the CPU and Memory benchmarks suffer from the most and the least variations, respectively. Interestingly, as we see in the following sections, commonly used performance evaluation methodologies (i.e., single trial and multiple consecutive trials) fail to provide valid comparisons between multiple alternatives even for the memory benchmark traces which are the most stable experiments.

Bench	Inst	Reg	Min	Max	Mean	SD	CV
CPU	S	US	55,444	122,831	103,643	23,918	0.23
		EU	55,670	123,010	108,890	19,526	0.18
	L	US	116,243	558,446	471,584	112,044	0.24
		EU	115,972	558,143	459,467	111,811	0.24
Mem	S	US	50,856	77,030	69,544	6,712	0.10
		EU	52,357	76,744	71,693	5,050	0.07
	L	US	119,696	320,642	290,211	31,345	0.11
		EU	181,566	321,758	293,390	28,472	0.10
Disk	S	US	7,972	84,267	59,551	11,875	0.20
		EU	10,489	78,693	61,757	8,388	0.14
	L	US	33,684	98,034	80,896	18,288	0.23
		EU	37,908	95,956	82,220	15,688	0.19
Net	M	US	158	836	579	117	0.20
		EU	378	886	720	88	0.12

Table 2: Traces statistical data. S: Small, L: Large, M: Mixed[1]

4.1 Single Trial

The single trial methodology is a very common approach for conducting experiments in cloud computing environments due to its simplicity. For example, 11 papers (out of 38) published in SoCC'16 utilize this technique. However, changes in the cloud environment may render this approach unreliable when comparing multiple alternatives. Figure 2 shows an example of how measured application performance can change from one trial to the next (each point is the benchmark score of a trial and should all be identical). Therefore, if comparisons are done using this methodology, it is not clear whether the observed differences are due to the difference in alternatives being studied or changes in the environment.

To better understand the performance variation between two consecutive trials, we compute the percentage difference between those trials (i.e., difference of consecutive trials divided by the minimum of these trials) for the CPU, memory, disk, and network traces collected for large instances located in the EU. In Figure 3, a value on the y-axis shows the ratio of consecutive trials that experience a difference less than or equal to the value on the x-axis for a given benchmark. For example, in the CPU benchmark, 20% of the consecutive trials experience more than a 100% change in performance. As depicted in the figure, for all traces, there are instances were the difference between two consecutive trials is more than 50%. Although the memory benchmark has the lowest variation, 17% of consecutive trials have a percentage difference of more than 20%. *As a result, a reliable comparison of multiple alternatives cannot be performed using the single trial methodology for any of the studied traces.*

4.2 Multiple Consecutive Trials

In the Multiple Consecutive Trials (MCT) methodology, experimenters often conduct all of the trials for a given alternative, before proceeding to the next. This is convenient, as there is generally some setup time involved in switching between alternatives (e.g., changing software configuration).

[1] Small and large instances use the same network interface, therefore, the instance size is irrelevant in the networking benchmarks.

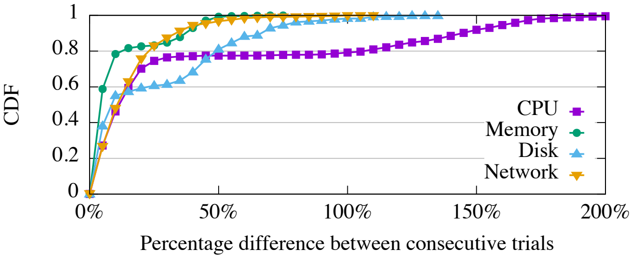


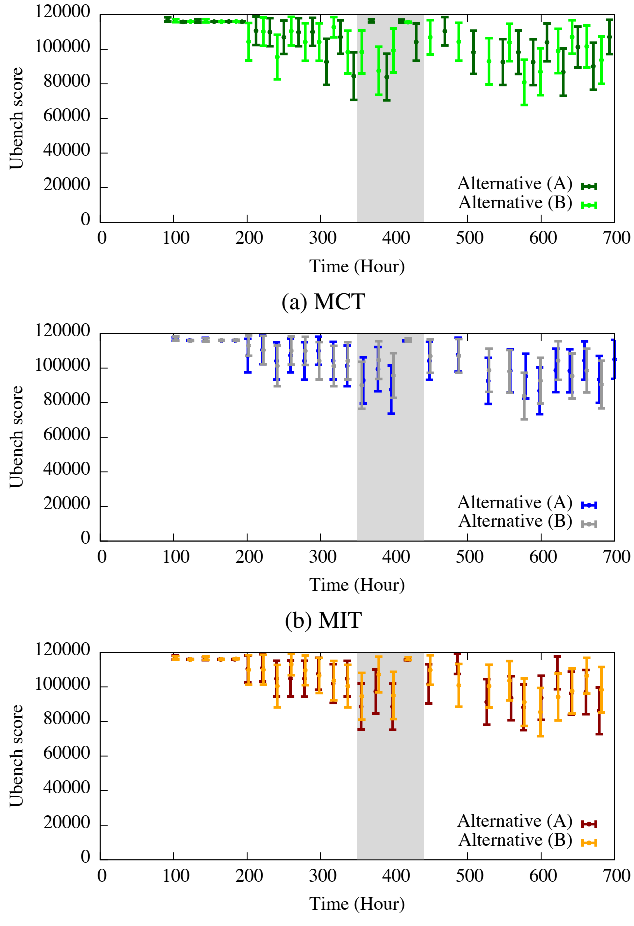
Figure 3: Single trial: falsely reported differences

To evaluate this approach, we combine 20 trials of a benchmark to constitute an experiment. For two alternatives, 20 trials for alternative *A* are run followed by 20 trials for alternative B and their results are compared. We compare results by computing the average of the 20 trials along with 95% confidence intervals. Since in each case *A* and *B* are identical benchmarks (two different labels for each alternative), a sound measurement methodology should show no statistical performance differences between them (i.e., no non-overlapping confidence intervals). However, if *any* performance difference is observed, it indicates that the measurement methodology might falsely associate the observed differences to differences in alternatives, while differences are rooted in the variable cloud computing environment. Therefore, the measurement technique can potentially lead to erroneous conclusions. When comparing the performance of alternatives *A* and *B*, we consider them statistically different if the confidence intervals *do not* overlap. On the other hand, the two alternatives are considered statistically similar if the confidence intervals *do* overlap.

Figure 4-a shows the results of applying the multiple consecutive trials technique on the CPU benchmark trace obtained in the US. The *x* and *y* axes represent the time and average of 20 benchmark scores, respectively. Note that Figures 4 and 5 are missing a few data points. There is no valid measurement score for these times in the provided trace (see [14]). The gray area highlights the time window where the non-overlapping confidence intervals are observed for one of the methodologies for a given trace. *Note that in practice only two average values are used to compare A and B. However, we continue applying MCT to obtain more points in order to observe what the outcome would be if the experiments were conducted at different times of the day.*

As seen in the figure, there are a few instances of non-overlapping confidence intervals between time 350 and 450 (highlighted with the gray box). In these cases, the alternative *A* is statistically different from the previous or next alternative *B* measurement. A similar problem is observed for the memory benchmark trace shown in Figure 5-a. Despite smaller variations in these measurements (i.e., smaller confidence intervals), there are a few non-overlapping confidence intervals for some consecutive experiments. Recall that *A* and *B* are identical, therefore, these statistically significant differences are a sign that the measurement technique is flawed. We repeated the CPU and memory speed experiments with 5 and 10 trials, instead of 20 trials, and similar results (i.e., non-overlapping confidence intervals) were observed.

The confidence intervals obtained using this technique may provide a false sense of rigor and validity when comparing different alternatives. The differences are in reality due to differences in the environment rather than the different alternatives.

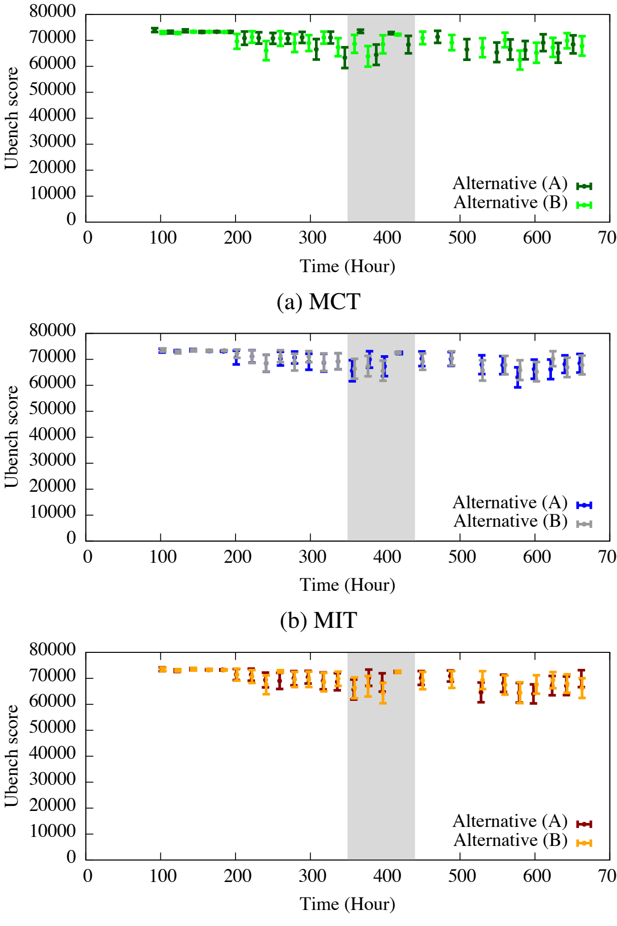


(a) MCT

(b) MIT

(c) RMIT

Figure 4: CPU performance, small instance, US.



(a) MCT

(b) MIT

(c) RMIT

Figure 5: Memory performance, small instance, US.

We repeat the same evaluation using all traces we introduced in Table 2 and summarize the results in Table 3. For each trace, we report the Ratio of Flawed Experiments to the total number of experiments (RFE) (presented in percentage). For instance if RFE = 10%, it means that 10% of comparisons have non-overlapping confidence intervals. Note that it is not possible to tell prior to or after running experiments whether or not the measurement is flawed, as a result any non-zero value for the RFE is not acceptable for a methodology. In Table 3, we also present the maximum difference incorrectly reported by a flawed methodology. This measure is the ratio of the difference between the two mean values to the minimum of the two mean values (also presented in percentage).

Table 3 shows that, except for two disk I/O traces, the ratio of flawed experiments (RFE) is at least 4% when using the MCT methodology, rendering this technique error-prone. The maximum difference metric is also alarming. In the CPU benchmark experiments, the MCT methodology falsely reports a difference of up to 37.8% as being statistically significant (with a confidence level of 0.95) for two identical alternatives. In other words, the MCT methodology determines that a system is 37.8% better than itself! In addition to these traces, we also applied the MCT methodology on a networking trace collected using Microsoft Azure's cloud computing environment in 2013 by the authors of [14]. We found that the networking benchmark had a RFE of 14.3% and max difference of 16.6%. Our findings demonstrate that the MCT methodology must be avoided when performing any performance evaluation in cloud computing environment.

Benchmark	Instance	Region	RFE	Max Diff
CPU	Small	US	10.2	37.8
		EU	9.8	27.3
	Large	US	7.9	30.1
		EU	14.9	36.5
Memory	Small	US	9.8	15.1
		EU	6.7	8.8
	Large	US	10.2	16.8
		EU	7.0	11.1
Disk I/O	Small	US	4.0	15.3
		EU	0.0	NA
	Large	US	5.9	23.8
		EU	0.0	NA
Network	Mixed	US	6.1	19.6
		EU	5.1	12.5

Table 3: Efficacy of the MCT methodology

4.3 Multiple Interleaved Trials

The intuition behind the Multiple Interleaved Trials (MIT) methodology is that by interleaving alternatives, over time, trials of the different alternatives are closer together in time. As a result, the alternatives will be exposed to conditions that are more similar than when using multiple consecutive trials.

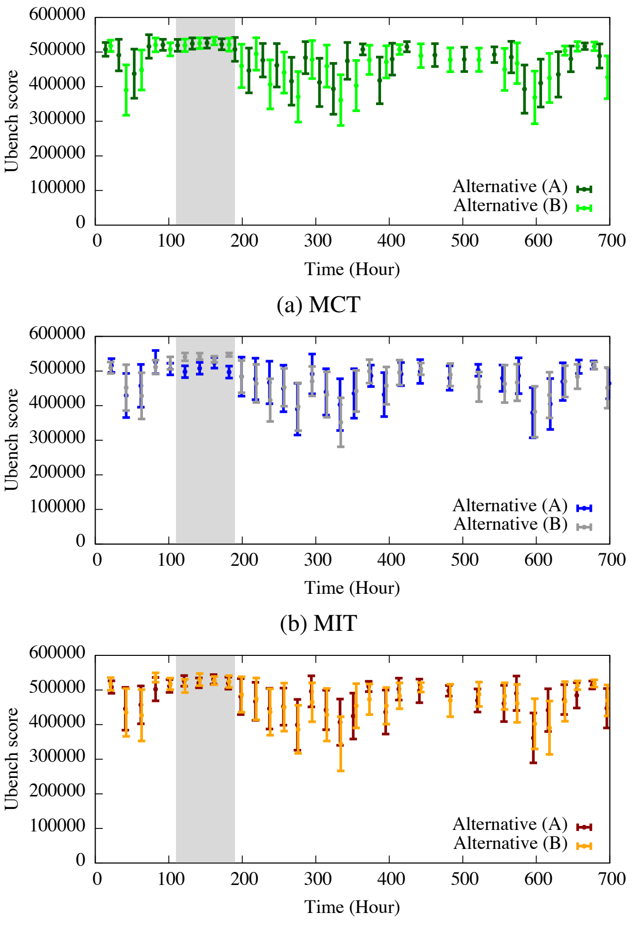
Figures 4-b and 5-b show the results of applying the multiple interleaved trials technique to the CPU and memory benchmark traces, in the hope of addressing the shortcomings of the MCT technique. As illustrated in these figures, the MIT methodology successfully avoids any non-overlapping confidence intervals. Therefore, based on the obtained results, alternatives *A* and *B* are statistically the same, as they should be.

As mentioned earlier, if the cloud computing environment is affected at regular intervals, it is possible that some alternatives could be affected more than others. We have found that scenarios do exist where periodic changes in the cloud computing environment adversely affect the multiple interleaved trials technique. Figure 6 shows how the multiple interleaved trials technique is unable to provide statistically similar results for alternatives *A* and *B*. A few instances of non-overlapping confidence intervals were observed between time 100 and 200.

To ensure that periodic changes cause this behavior, we plot the raw measurement data between time 100 and 180 (roughly corresponding to the shaded area). If two alternatives are compared using the MIT technique, the performance of each alternative is measured using only odd or even trials. In Figure 7, which shows the raw measurement data, we mark the odd and even trials differently in an attempt to find a particular pattern that explains the non-overlapping confidence intervals. As seen in this figure, a periodic pattern is observed for this particular time window which causes odd trials to have a lower performance score in general. As a result, 3 experiments in this time window (shaded area in Figure 6-b) obtain non-overlapping confidence intervals. The only exception is the time window between 140 and 160 where there is no obvious superior or inferior performance for the odd and even trials. In Figure 6-b, the corresponding experiments at time 160 (the second experiment from the right in the shaded area) obtains overlapping confidence intervals. *This demonstrates that trials must be randomized to avoid such problems.*

4.4 Randomized Multiple Interleaved Trials

As we saw in Section 4.3, the multiple interleaved trials methodology can not be used if the environment changes occur regularly with a period that overlaps with the length of the trials. Therefore, it is critical to randomize the order of trials to avoid this problem.



(a) MCT

(b) MIT

(c) RMIT

Figure 6: CPU performance, large instance, US.

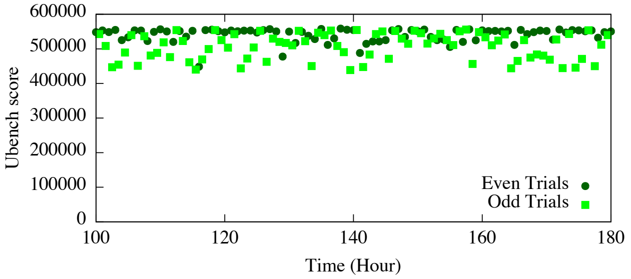


Figure 7: CPU performance, large instance, US.

The Randomized Multiple Interleaved Trials (RMIT) technique reorders the trials randomly during each round to avoid ensure that the same alternatives are not affected by periodic changes in the cloud computing environment.

We apply the RMIT methodology to the CPU performance benchmark and the results are shown in Figure 6-c. A comparisons between Figure (c) and (b) shows that the RMIT technique produces no non-overlapping confidence interval and correctly shows that there is no statistical difference between alternatives *A* and *B*. Figures 4-c and 5-c, also show that comparisons between alternatives *A* and *B* are statistically correct.

These results show that, RMIT can be used in highly variable cloud environments to compare multiple alternatives and that this methodology inherently captures the variability in performance metrics across runs. We note that if the variations are large, it may not be possible to distinguish relatively small differences between two or more alternatives. The applicability of this technique depends on the variations in the environment and the significance of the difference between alternatives.

Note that it is not possible to tell prior to or after running experiments whether or not the performance metrics may have been affected by changes in the cloud computing environment, nor if changes in the cloud computing environment may be periodic. Therefore, it is critical that randomized multiple interleaved trials be used when comparing alternatives.

5. DISCUSSION

In this work, we focus on the sequential execution of trials on a single VM. However, it is possible to run multiple trials in parallel on multiple VMs. Suppose that we compare N alternatives which require M VMs each ($M > 1$ means we are running a distributed application). All of the N alternatives are assigned to a total of $N \times M$ VMs. Because the $N \times M$ VMs might not be identical, and because some VMs might be affected by other applications that are running on the same physical hosts or on the same physical rack, one can randomly reassign the alternatives to the VMs for each trial. This procedure can be continued for the desired number of trials. As a result, different alternatives will all experience the same conditions. The complete design and evaluation of such a methodology is left for future work.

6. CONCLUSIONS

Using cloud computing environments for experimental performance evaluations is fraught with difficulties because of the possibly highly variable nature of the environment. We utilize CPU, memory, disk, and network performance traces obtained from Amazon EC2 servers and find that commonly used methodologies for conducting experiments like using single trial or multiple consecutive trials could lead to erroneous conclusions. We found that these methodologies are used by many papers published in the SoCC'16 conference. However, we do find that the randomized multiple interleaved trials methodology can be used to obtain repeatable performance metrics and should form a basis for the fair comparison of competing alternatives by appropriately randomizing the time at which trials of each alternative are conducted.

7. ACKNOWLEDGMENTS

Funding for this project was provided in part by a Natural Sciences and Engineering Research Council (NSERC) of Canada Discovery Grant and a Discovery Accelerator Supplement. We also thank Kamal Rahimi Malekshan for his careful proofreading of the paper and the anonymous reviewers for their helpful feedback. We are especially greatful to Jörg Schad, Jens Dittrich, and Jorge-Arnulfo Quiané-Ruiz for providing us with their data and allowing us to make it available to others.

8. REFERENCES

[1] http://cs.uwaterloo.ca/˜brecht/data/icpe-rmit-2017.

[2] Abedi, A., Heard, A., and Brecht, T. Conducting repeatable experiments and fair comparisons using 802.11N MIMO networks. *SIGOPS Oper. Syst. Rev. 49*, 1 (2015).

[3] Ballani, H., Costa, P., Karagiannis, T., and Rowstron, A. Towards predictable datacenter networks. In *SIGCOMM* (2011).

[4] Ballani, H., Jang, K., Karagiannis, T., Kim, C., Gunawardena, D., and O'Shea, G. Chatty tenants and the cloud network sharing problem. In *NSDI* (2013).

[5] Cardosa, M., Wang, C., Nangia, A., Chandra, A., and Weissman, J. Exploring MapReduce efficiency with highly-distributed data. In *MapReduce* (2011).

[6] Farley, B., Juels, A., Varadarajan, V., Ristenpart, T., Bowers, K. D., and Swift, M. M. More for your money: Exploiting performance heterogeneity in public clouds. In *SoCC* (2012).

[7] Iordache, A., Morin, C., Parlavantzas, N., Feller, E., and Riteau, P. Resilin: Elastic MapReduce over multiple clouds. In *CCGrid* (2013).

[8] Iosup, A., Yigitbasi, N., and Epema, D. On the performance variability of production cloud services. In *CCGRID* (2011).

[9] Jiang, D., Ooi, B., Shi, L., and Wu, S. The performance of MapReduce: An in-depth study. *Proc. VLDB Endow. 3*, 1-2 (2010), 472–483.

[10] Juve, G., Deelman, E., Berriman, G. B., Berman, B. P., and Maechling, P. An evaluation of the cost and performance of scientific workflows on Amazon EC2. *J. Grid Comput. 10*, 1 (2012).

[11] Leitner, P., and Cito, J. Patterns in the chaos - a study of performance variation and predictability in public iaas clouds. *ACM Trans. Internet Technol. 16*, 3 (2016).

[12] Mehrotra, P., Djomehri, J., Heistand, S., Hood, R., Jin, H., Lazanoff, A., Saini, S., and Biswas, R. Performance evaluation of Amazon EC2 for NASA HPC applications. In *ScienceCloud* (2012).

[13] Montgomery, D. *Design and Analysis of Experiments*. Wiley, 2012.

[14] Schad, J., Dittrich, J., and Quiané-Ruiz, J.-A. Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance. *Proc. VLDB Endow.* (2010).